Schell85 Schell, R., Tao, T., and Heckman, M., "Designing the GEMSOS Security Kernel for Security and Performance," *Proceedings of the 8th National Computer Security Conference*, 1985.

This paper reports on the major design choices for security functionality and reports the results of initial system performance measurements on the Version 0 GEMSOS commercial product.

Other Readings

Ames83 Ames, S.R., Gasser, M., and Schell, R., "Security Kernel Design and Implementation: An Introduction," *Computer*, Vol. 16, No. 7, pp. 14-25, July 1983.

This paper provides an excellent overview of the security kernel and some underlying principles and considerations for security kernel design and implementation.

Grenier89 Grenier, G., Holt, R., and Funkenhauser, M., "Policy vs. Mechanism in the Secure TUNIS Operating System," *Proceedings of the 1989 IEEE Symposium on Security and Privacy*, May 1989.

The TCB of a secure operating sytem (OS) can have its security policy enforced by a small, provably correct security manager. The design of the Secure TUNIS OS divides security concerns into policy (implemented by the security manager) and mechanism (implemented by the rest of the OS). Secure TUNIS is targeted for B3 and above.

Roberts85 Roberts, P.M., *Data Security: A Growing Concern*, Honeywell Information Systems, 1985.

This is a survey paper about the need for data security. It mentions the reference monitor and SCOMP.

Schell83 Schell, R.R., "A Security Kernel for a Multiprocessor Microcomputer," *Computer*, Vol. 16, No. 7, pp. 47-54, July 1983.

This paper talks about the implementation of a security kernel for a multiprocessor microcomputer which faced two specific challenges: (1) providing adequate computational resources for applications tasks and (2) developing a clean, straightforward structure whose correctness could be easily reviewed.

C2:

15. Describe the interfaces (control and data flow) among the TCB elements.

16. Describe the interface between the kernel and the rest of the TCB elements.

B1:

19. (a) List software mechanisms that are used to isolate and protect user processes. (b) Provide a brief description of eachmechanism.

B2:

23. (a) For each TCB element, identify protection-critical portions of the code. (b) Describe the protection-critical functions performed by the code.

B3:

27. (a) For each TCB element, identify non-protection-critical portions of the code. (b) Explain why the code is part of the TCB.

Required Readings

TCSEC85 National Computer Security Center, *Department of Defense Trusted Computer System Evaluation Criteria*, DoD 5200.28-STD, December 1985.

Section 6.1 gives a brief introduction to the reference monitor concept, and Section 6.3 introduces the TCB. Sections 2.1.3.1.1, 2.2.3.1.1, 3.1.3.1.1, 3.2.3.1.1, 3.3.3.1.1 and 4.1.3.1.1 contain the primary TCB requirements, which are summarized on page 105. NOTE: These system architecture requirements are addressed in much more detail in Module 7. They should be examined in this module to get a feeling for what is required of the TCB. Module 7 describes architectures that can meet these requirements.

Gasser88 Gasser, M., *Building a Secure Computer System*, Van Nostrand Reinhold Co., N.Y., 1988.

Chapters 4.2 and 10.1-10.3 talk about the reference monitor. Chapter 3.2 talks about the TCB and the TCB perimeter, while all of Chapter 10 talks about the concept of a security kernel.

Supplemental Readings

Saydjari87 Saydjari, O.S., Beckman, J.M., and Leaman, J.R., "Locking Computers Securely," *Proceedings of the 10th National Computer Security Conference*, September 1987.

LOCK is a technology research and development project to build a hardware-based reference monitor module. This module will be generic and thus reusable on many different computers. Full advantage will be taken of inexpensive generic cryptographic modules currently in development.

exceeded. If the occurrence or accumulation of these events continues, the mechanism will terminate the event using the action least disruptive to the system.

Isolation:

No additional requirement.

Verifiability:

The TCB uses a conceptually simple protection mechanism that is complete and has precisely defined semantics. This protection mechanism plays a central role in determining the internal structure of the TCB.

The internal structure of the TCB is modular, and the design makes significant use of data hiding, abstraction and layering.

The size and complexity of the TCB are minimized, rigorously excluding functionality that is not protection-critical.

A1 Reference Monitor Requirements

Completeness:

No additional requirement.

Isolation:

No additional requirement.

Verifiability:

No additional requirement.

Relevant Trusted Product Evaluation Questionnaire Questions

Some of the TPE questions refer to “protection-critical” code. This is the code that directly enforces the system security policy by managing and implementing the access control mechanisms.

2.4. SOFTWARE

The TCB software consists of the elements that are involved in enforcing the system security policy. Examples of TCB elements include: kernel, interrupt handlers, process manager, I/O handlers, I/O manager, user/process interface, hardware diagnostics, hardware exercisers, and command languages/ interfaces (for system generation, operator, administrator, users etc.). The security kernel is the hardware, firmware and software elements of the TCB that are involved in implementing the reference monitor concept, i.e., the ones that mediate all access to objects by subjects.

C1:

1. Provide a (a) description and (b) architecture of theTrusted Computing Base (TCB) at the element level (see above examples for elements).
2. Describe the interface between the TCB and user processes that are outside the TCB.

B1 Reference Monitor Requirements

Completeness:

The TCB maintains sensitivity labels on all subjects and storage objects under its control, and uses the labels as the basis for MAC decisions.

Isolation:

The TCB isolates processes by controlling their (distinct) addressspaces .

Verifiability:

No additional requirement.

B2 Reference Monitor Requirements

Completeness:

The TCB maintains sensitivity labels on all system resources that are directly or indirectly accessible by non-TCB subjects, and uses the labels as the basis for MAC decisions.

The TCB maintains minimum and maximum sensitivity labels for each device, controlling the range of levels of information that can be processed by the device.

The design of the TCB enforces the principle of least privilege.

The user interface to the TCB is completely defined.

Isolation:

Features in the hardware exist that allow the TCB to isolate separate objects with separate attributes (*e.g.*, readable writeable, executable), and, along with any other useful features of the hardware base, they are effectively used to isolate protection-critical portions of the TCB.

Verifiability:

The TCB shall be completely and accurately described in terms of exceptions, error messages, and effects.

The internal structure of the TCB is modular.

B3 Reference Monitor Requirements

Completeness:

The TCB provides a finer granularity of DAC which supports specific access modes (including one that denies access) for individuals *and* groups.

The TCB contains trusted recovery mechanisms that ensure that a protection compromise does not occur during recovery from a system failure or other discontinuity.

The TCB contains a mechanism that monitors the occurrence or accumulation of events that may indicate an imminent violation of the security policy (e.g., the exploitation of a covert channel). This mechanism will immediately notify the security administrator when thresholds are

Isolation: Mechanisms or design features of the TCB that help to prevent non-TCB subjects from interfering or tampering with the operation of the TCB. Life-cycle assurance requirements with regard to configuration management and trusted distribution are omitted because the reference monitor concept of isolation is treated here as features that help arunning TCB protect itself, not how developers protect stored copies of the components of the TCB.

Verifiability: Design features and strategies that help make the TCBsmall and understandable. Documentation and testing requirements are part of the effort to actually verify the correctness of the TCB, as opposed to an effort to make the TCB small and understandable, so they are omitted.

Note that some of the TCSEC requirements meet the above mentioned parameters for categorization as supporting more than one of thereference monitor concepts. These requirements are not duplicated but are categorized as supporting the reference monitor concept that seemed most appropriate.

C1 Reference Monitor Requirements

Completeness:

The TCB defines and controls access between named users (or groupsof users) and named objects (i.e., DAC).

Isolation:

The TCB maintains an execution domain that protects it from tampering by non-TCB subjects.

Verifiability:

No additional requirement.

C2 Reference Monitor Requirements

Completeness:

The TCB provides a finer granularity of DAC (including or excluding a single user).

The TCB provides controls to limit the propagation of access rights.

Access to an object can only be given by an authorized user.

Isolation:

The TCB isolates the resources to be protected, in order to controlaccess to the resources.

Verifiability:

No additional requirement.

In addition to the trusted portions of the operating system itself, any software outside of the operating system that must be trusted to uphold thesecurity policy is also included in the TCB. This software usually comes in the form of trusted subjects (e.g., trusted processes) that have been given privileges under very controlled conditions. Trusted subjects may bypass the security mechanisms of the system, but are trusted not to violate the securitypolicy . These subjects must be trusted to perform their function correctly. For example, a trusted process that is used to downgrade information must be trusted to only reclassify information that has been designated by a cleared user with authority to do so. Another example would be a trusted printer spooler. When users query the print spooler as to the status of their printjobs , the spooler must be trusted not to reveal any information about higherlevel print jobs that the spooler may have previously seen. In this case, the spooler process is not being trusted as a downgrader, but it is trusted not to reveal information about any print jobs more sensitive than the user's current sensitivity label.

The TCB must be able to protect itself from modification by untrustedsubjects . This protection typically comes in the form of some type of hardware based memory protection that isolates untrusted subjects from each other as well as from the TCB. Hardware mechanisms must isolate untrusted subjects from each other and subjects from objects except when the TCB determines that access is authorized under the system security policy. Only through TCB calls can a subject request access to an object. The TCB then grants or deniesthe request based on the subject's privileges. If an access request is granted, the TCB performs whatever actions are necessary to configure the hardware protection mechanism to allow access to the requested object.

Reference Monitor Requirements Breakdown

To get a better feel for the requirements placed on a reference monitor bythe TCSEC, the relevant requirements at each assurance class have been subdivided below among the three reference monitor requirements for completeness, isolation, and verifiability. The requirements were extracted from Appendix D of the TCSEC. The summaries should be read sequentially, because the requirements for each TCSEC class include the requirements of the lower classes. The abbreviation NAR stands for "no additional requirement" and NR stands for "no requirement." The TCSEC requirements were categorized with regard to the three reference monitor conceptsusing the following parameters:

Completeness: Mechanisms or design features of the TCB that help to prevent the TCB from being bypassed. This implies that all access to information be mediated by the kernel consistent with the definedsecurity policy. Included are aspects of access control that specify to the TCB how subjects may access objects. I&A requirements are omitted because the reference monitor completeness concept is treated here as preventing bypass of the TCB by subjects that are assumed to have been identified and authenticated. Audit requirements are omitted (with one exceptionat B3) because this concept refers to preventing bypass of the TCB, not reporting that it may be happening.

By developing a security kernel as the central core of the operatingsystem, security relevant software responsible for access control decisions is segregated into a single trusted core that is much smaller than the rest of the operating system. This helps the security kernel satisfy the three requirementsplaced on a reference monitor; that it be tamperproof, impossible to bypass, and verifiable. The security kernel must be protected from tampering to ensure its correct operation. It must not be possible to bypass its access controlc hecks, otherwise the security policy could be violated. Finally, the security kernel must be verified to ensure that it enforces the security policycorrectly and will always perform the proper access control checks. The small size of the security kernel facilitates the process of verifying or proving that the referencemonitor enforces the access control policy. Verification that a kernel is correct may, depending upon the degree of assurance sought, be accomplished by such techniques as code inspection, thorough testing, or formal mathematical specification and verification.

Formal verification has been found to be more difficult than expectedand formal verification technology has been slow to mature. To formally verify a kernel requires proving aspects of correctness about a large program.This is a difficult problem. Of the different stages of kernel development towhic h verification can be applied, specification verification andcorrespondence proofs against the model of the system have been found to offer the greatest degree of success. Code level verification and correspondence against thespecification must be further investigated. Even without a full mathematical proof ofa system, adherence to the rigorous review and documentation generation required by a verification methodology will work to instill much more trust into the end product. More detail about security policy modeling andverification is presented in Module 5 and Module 13, respectively.

Trusted Computing Base

A TCB comprises all of the hardware and software that must be trusted to enforce the system security policy. This includes access mediation software as well as the identification and authentication (I&A) mechanism, the audit mechanism, and software which cannot be constrained by the rules enforced for untrusted software. Any part, up to and including the entire operating system, must be considered to be in the TCB if it must be trusted not to violatethe security policy. The TCB of a TCSEC class B2 system should be internally structured into well-defined largely independent modules, and elements that are protection-critical should be separated by hardware from those that are not. For highly trusted systems (B3 and higher) the TCB must beminimized as much as possible. Most of the operating system for highly trusted systems must be outside of the TCB. However, the TCB must include all trusted software while being isolated from everything outside the TCB. If software has the ability to affect the execution of the TCB, it must be included in the TCB, even though it may not be security critical. Everything inside of the TCB must be completely protected from everything outside of the TCB while thecontents of the TCB must be trusted to enforce the system security policy and notinterfere with other portions of the TCB.

that all security-relevant software is segregated into a trusted kernel. Centralizing access control into an RVM, such as a “security kernel,” simplifies the necessary tasks of ensuring that these controls may not be tampered with nor bypassed. By making such a kernel small, the correctness of what it enforces is easier to verify. A security kernel is entrusted with controlling all accesses by subjects (users or processes) to objects (files, directories, etc.), and thus must handle the parts of an operating system that manage resources(e .g., memory, disk space) shared by multiple users. When an access is requested, the security kernel determines whether the operation is allowed by thesecurity policy and, if it is allowed, grants the access. It should be noted that there may be other ways to build a system that satisfies reference monitor concepts, but no other approach is as well-developed as the security kernel.

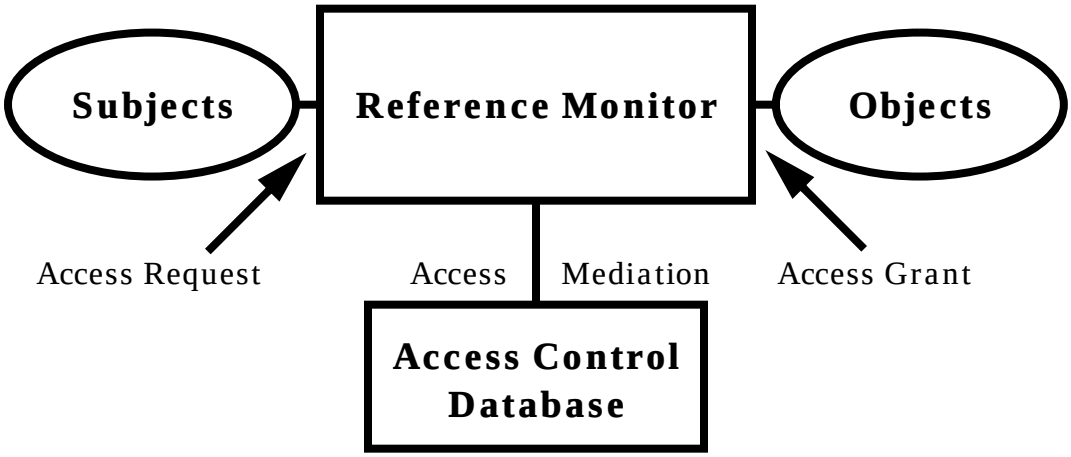

Figure 6-1. The Reference Monitor

Trade-offs must often be made among performance, functionality, convenience, and complexity in determining whether any non-security-related functionsare included in the kernel. When the trusted system is emulating a previously existing operating system, functions may be especially difficult to reallocate to cleanly separate security-related from non-security-related functions. There is nothing explicitly preventing non-security-related functions from being included in the kernel, but doing so will directly conflict with the goalof making the kernel as small as possible so that its correctness is easy to verify.

In a discussion of security kernel design and implementation, [Ames83] identifies four general architectural areas in which specific hardware and software mechanisms have proved useful or necessary to support a kernel-based general purpose operating system: explicit processes to provideefficient support for multiprogramming and interprocess communication, memory protection, execution domains (e.g., user, supervisor, and kernel), and I/O mediation. Many modern computer architectures provide the necessary underlying hardware features, but significant performance degradation may occur without further modification to the hardware design to directly address the requirements of a kernel-based architecture.

Reference Monitor and Trusted Computing Base

This module describes the concepts of reference monitor and trustedcomputing base (TCB) and examines their relationship to the security policythey enforce .

Module Learning Objectives

The material presented in this module describes the mechanisms that can be used to enforce the policies and policy models described Module 5. Upon completion of this module, the student should:

1. Understand what a reference monitor is.
2. Understand what requirements a reference monitor must satisfy.
3. Understand what a TCB is.
4. Understand where the TCB perimeter lies.
5. Understand the difference between a trusted and untrusted subject.

Overview

The previous module talked about security policies and security policymodels . A system security policy defines what is meant for a system to besecure -- a system is “secure” only with respect to some specific definedpolicy . A security policy model formalizes the security policy and describes rules ofoperations that the security enforcement mechanisms regulate. The reference monitor concept refers to the abstract machine enforcing all access mediation between subjects and objects. Subjects present access requests, and the reference validation mechanism (RVM) implementing the reference monitor mediates the requests according to information from an abstract access controldatabase . The access control database embodies the security state of the systemand suc h dynamic information as the security attributes of the objects and theaccess rights of the subjects supported by the system. If the desired access is consistent with the security policy, then the RVM grants a subject the requested access to the appropriate object. This scenario is depicted inF igure 6-1.

A TCB is the combination of hardware and software that is responsible for enforcing the security policy of the system. It includes the RVM, as well as all other security critical hardware and software. The interface to the TCB consists of one or more of the following: untrusted user interface, trusted user interface (e.g., system security administrator interface), untrusted subject interface, trusted subject interface, external network interface. The TCB enforces the mandatory and discretionary access control (MAC & DAC)policy for the system, as defined by the security policy. The exact MAC and DAC checks that must be made are described in the rules of operation in thesecurity policy model.

Reference Monitor Implementation

The security checks performed by an RVM can be distributed throughout an operating system (which is what is typically done by untrusted operating systems) or the attempt can be made to restructure the operatingsystem so